Primm Skills assessment

David Scott Primm

April 20, 2020

Contents

1 Project overview

This is a skill assessment from Novetta
Write a network server and client for cryptographically hashing files. The hashing server will be listening on the network. The client will connect to the server, indicate the desired hashing algorithm, upload the file, and receive the hash of the file. The client will then display the hash.

2 hashclient

hashclient takes arguments from the command line, and sends its specified files to hashserver, receiving the hash value in return from each file sent

2.1 requirements

- The client must be written in C.
- The client must compile using gcc on Ubuntu 16.04 or 18.04.
- The client will be executed on Ubuntu 16.04 or 18.04.
- The client must accept the following arguments:

  Port: The TCP port to connect on. This argument is optional.

  Address: The IP address where the server is listening. This argument is mandatory.

  Hash name: The name of the hashing algorithm to use. The following values should be supported: sha1, sha256, sha512, and md5. This argument is mandatory.

  Filename: The name of the file or files to hash. A minimum of one filename is required with no maximum.

  Example usage: hashclient [-p port] ¡address¿ ¡hashname¿ ¡filename¿...
- The client will print the name and hash (in hex format) of each file specified to stdout.

2.2 usage

```
hashclient [port] <hash> <IP> <file...>
[port] is optional, it will default to 2345
<hash> is the requested hash type to be used, avaliable options are md5,
     sha1, sha224, sha256, sha384, sha512, shake_128, shake_256, blake2b
<IP> is the IP address of the hashserver
<file...> is the list of space separated files to be hashed
```

2.3 example usage

```
./hashclient md5 127.0.0.1 makefile
sending makefile
ae618bad09eca643e57480ef3922a98e makefile
```

2.4 issues

If hashclient is ran with no files, it will simply send a termination message to the hashserver.
If hashclient sends the same file multiple times, on occasion the hashserver crashes, the cause was not found.

3 hashserver

hashserver takes arguments from the command line, and begins listening for a connection on its specified port, on connection it writes the received file out to a filename made from calling time.time, it then hashes the received file, and sends the hash back to the hashclient

3.1 requirements

- The server must be written in python.
- The server will be executed using python 3.
- The server must accept the following arguments:

  Port: The TCP port to listen on. This argument is optional.

3.2 usage

```
hashserver.py [port] [IP]
[port] is optional, it will default to 2345
[IP] is the IP address hashserver will bind to, defaults to 127.0.0.1
```

3.3 example usage

```
python3 hashserver.py
waiting for connection
connected to 127.0.0.1
waiting for connection
```

3.4 issues

hashserver is designed to run in python3 only, it should terminate if ran in an older version of python
hashserver sometimes crashes when receiving the same file multiple times, the cause of this was not found.